APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/750,517	12/31/2003	Michael G. Lisanke	SOM920030007US1	9240

55420 7590 05/12/2008
FLEIT, KAIN, GIBBONS, GUTMAN, BONGINI & BIANCO P.L
551 NW 77TH STREET
SUITE 111
BOCA RATON, FL 33487

EXAMINER	
WANG, HARRIS C	

ART UNIT	PAPER NUMBER
2139	

NOTIFICATION DATE	DELIVERY MODE
05/12/2008	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

ptoboca@focusonip.com

Office Action Summary	Application No.	Applicant(s)	
	10/750,517	LISANKE ET AL.	
	Examiner	Art Unit	
	HARRIS C. WANG	2139	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE _3_ MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on _02 January 2008_.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) _1-22_ is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) _1-22_ is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.	6) ☐ Other: _____.

DETAILED ACTION

Response to Arguments

The Applicant writes “The entire purpose of Circenis is to detect any tampering of a file by a user. Therefore, if a user was given the ability to turn on/off logging as recited for the presently claimed invention, the tamper-evident management system in Circenis would be defeated." However, nowhere in claim 1 does the Applicant claim the ability to turn on/off logging, only that the end user identifies and selects an instance to be monitored by the processor. Therefore the previous argument is considered spurious.

The Applicant next argues that “The monitoring of the presently claimed invention is only performed on the instances selected by the end user (pg. 11-12).” While the Applicant does claim “wherein the one instance is identified and selected by an end user to be monitored,” the Applicant never claims that the monitoring is <u>only</u> performed on the instances selected by the end user. Therefore the previous argument is considered spurious.

The Applicant then argues that “The Applicants are unsure of how the Examiner concluded that Circenis teaches “whereby the set of data is used to diagnose the software execution. (pg. 12 of Remarks)” Circenis uses the data to diagnose whether the data has been tampered with or not.

The Applicant's arguments regarding Claim 7, are moot in view of the new amendment. The Applicant agrees "The Applicant respectfully suggest that this argument made by the Examiner now fails in view of the amendment made (pg. 13 of Remarks)."

The Applicant then argues that "The Applicants are unsure how the Examiner concludes that a circular file, as taught by the IBM reference is <u>always populated with random data when created</u> (pg. 15 of Remarks)."

The Examiner never argues that it is inherent that the circular file is always populated with random data when created, only that the file is populated with <u>some</u> data when created. Whenever a file is created, of any size, some value is inside the file. Even in a blank entry, some value exists inside the file.

The Applicants amendments are anticipated by the previously cited art, therefore the Applicant's arguments are found to be unpersuasive.

Claim Rejections - 35 USC § 112

Claims 1-22 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the written description requirement. The claim(s) contains subject matter which was not described in the specification in such a way as to reasonably convey to one skilled in the relevant art that the inventor(s), at the time the application was filed, had possession of the claimed invention. Regarding Claims 1, 10, 15, 20, the Applicant has amended to include the limitations: <u>wherein the one instance (of software execution) is</u> identified and selected by an <u>end-user</u> to be monitored <u>by the processor, wherein the end-user is a</u>

user that initiates execution of the software at a system associated with the end-user, and wherein the processor creates a log entry...in response to the one instance being identified and selected to be monitored.

The Applicant writes "the presently claimed invention, on the other ,(sic) hand, is only monitoring a specific instance of a software execution that has been selected by the end user (pages 10-11 of Remarks)."

However, the Examiner has searched the Applicant's specification and has not found any evidence that the end user "selects" an instance to be monitored. In fact, many parts of the Applicants specification directly state that the user can not even determine what is being logged. Paragraph [0019] of the Applicants specification reads "The present invention also has the advantage that the...content of messages logged are all obscured from the user of the system on which the logging is done....it is more difficult for the user to find the logging and to determine what is being logged."

It is unclear how the end user can select an instance of software execution to be monitored, when the end user has no control over what is being logged. The Applicant did not provide any citations of the Specification that support the new amendments..

Claims 2-9, 11-14, 16-19, 21-22 depend on the above claims and are rejected for the same rationale.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the

invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

Claims 1-22 are rejected under 35 U.S.C. 103(a) as being unpatentable over Circenis (US 20040054908) in view of the OpenPGP standard (RFC 2440) further in view of The IBM Certification Study Guide AIX V4.5 System Administration (1999) (hereafter referred to as IBM).

Regarding Claims 1 and 3,

Circenis teaches a system that allows analysis of software running in a tamper-resistant environment, the system comprising *("A tamper-evident data management system…includes an application for collecting usage or metrics data from the computer system" Abstract).:*

a processor which monitors at least one instance of software execution, wherein the one instance is identified and selected by an end-user to be monitored by the processor, wherein the end-user is a user that initiates execution of the software at a system associated with the end-user is a user that initiates execution of the software at a system associated with the end-user and wherein the processor creates a log entry with at least one of a set of data is used to diagnose the software execution in response to the one instance being identified and selected to be monitored; *("Using the tamper-evident system 200 of Fig. 3, a sender is able to monitor and control application utilization by collecting data associated with the application, creating tamper-evident data records, and providing the tamper-evident data records" Paragraph [0037])*

an encryption system which encrypts the log entry for the at least one set of data *(Figure 4 teaches encrypting the log entry for at least one set of data, particularly **step 320** "Sign data entry with application private key", **step 325** "Encrypt with vendor public key" and **step 330** "Store in data log")*

The Examiner interprets the processor that monitors software execution, wherein the one instance is identified by an end-user to be monitored by the processor as the user executing software on a device wherein each instance of execution is to be monitored by the processor, and in response to the one instance being executed a log entry is created (See Paragraph [0019] where Circenis teaches the pay per use application collecting metrics data to a data log).

Circenis does not explicitly teach an encryption system which generates at least one symmetric key and encrypts the log entry for the at least one set of data using the symmetric key, wherein the encryption system encrypts the symmetric key using a public key associated with the encryption system, wherein the log file includes the symmetric key which has been encrypted with the public key.

PGP ("Pretty Good Privacy") is a program that provides cryptographic privacy and authentication, and was created by Phillip Zimmermann in 1991. The OpenPGP standard (1998) is cited, but any PGP product teaches the generic method of:

1. Creating a message
2. Generating a symmetric key to be used as a session key for the message
3. Encrypting the session key using each recipient's public key. These "encrypted session keys" start the message.
4. The sending PGP encrypts the message using the session key, which forms the remainder of the message.
5. The receiving PGP device decrypts the session key using the recipient's private key
6. The receiving PGP decrypts the message using the session key.

Because Circenis already teaches one method of encrypting the data log, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the public-private key encryption of Circenis with the well known method of PGP, where the symmetric key is generated, the log entry is encrypted using the symmetric key, a public key encrypts the symmetric key, and the log file includes the symmetric key which has been encrypted with the public key.

The motivation is that PGP provides a more secure way of encrypting the log entries.

Circenis and OpenPGP do not explicitly teach a

a log file of a relatively-fixed size which stores the log entry for the at least one set of data which have been encrypted;

IBM teaches

a log file of a relatively-fixed size which stores the log entry for the at least one set of data which have been encrypted; *("The alog command can maintain and manage logs. It reads standard input , writes to standard output, and copies the output into a fixed-size file. This file is treated as a circular log" Section 2.4.1)*

a system for wrapping around and filling the log file from a beginning when the log file has been filled, allowing the log file to remain at a substantially-constant size even after the log file has been filled with data and a new entry is received. *("If the file is full, new entries are written over the oldest existing entries" Section 2.4.1). It is inherent that a circular log will wrap around and fill the log file from a beginning when the log file has been filled.*

It would have been obvious to one of ordinary skill in the art at the time of the invention to modify the monitoring system of Circenis to store the encrypted log entries in a circular log as described by IBM.

The motivation is that a circular log is a well known way to store a log file, where the circular log is inherently of a fixed size. It is inherent that a circular log will contain at least a pointer which identifies the next storage location for a next log entry.

The combined references of Circenis and IBM do not explicitly teach where random data in the log file when it is originally created and which is replaced by log entries so that a size of the log including log entries appears to be a substantially-constant size;

It would have been obvious to one of ordinary skill in the art at the time of the invention to insert random data into the log file when it is initially created.

The motivation is to initialize the circular log.

Regarding Claim 2,

The combined references of Circenis, OpenPGP and IBM teach a system including the elements of claim 1 wherein the system includes a transmission system for sending the log file, upon command, to a secure processing location away from the system in which the log file was created. *("The data log may also be transmitted to a remote system (comprising, for example, the validation computer 150) over a network connection" Paragraph [0043] of Circenis, Figure 3 shows the transmission of the log file 115 to the secure processing location away from the system 150 Circenis)*

Regarding Claim 4,

The combined references of Circenis, OpenPGP and IBM teach a system including the elements of claim 1 wherein the system includes a mechanism for obscuring a log entry which has been created. *(Figure 4 of Circenis teaches encrypting the log entry for at least one set of data, particularly **step 320** "Sign data entry with application private key", **step 325** "Encrypt with vendor public key" and **step 330** "Store in data log")*

Regarding Claim 5,

The combined references of Circenis, OpenPGP and IBM teach a system including the elements of claim 4, Circenis further teaches the mechanism for obscuring the activity for which a log entry is created includes a printing function for writing into the log file.

("The customer site that forbids electronic media leaving the site may require that the vendor print out any validated and decrypted data logs and bring the printout back to the vendor site for processing and billing." Paragraph [0034] *Circenis*) The Examiner interprets printing out the data logs as the printing function.

Regarding Claim 6,

The combined references of Circenis, OpenPGP and IBM teach a system including the elements of claim 2 wherein the system includes a mechanism for receiving an indication from a user that transmission is desired and transmits the log file in response to that indication. *("Fig. 5 is a flowchart illustrating steps in validating the data. The program starts (step 355) and the data log is copied to the validation computer* *through an intermediary device or medium (step 360)" Circenis) Before the data can be validated there must inherently be some indication for the log file to be transmitted.*

Regarding Claim 7,

The combined references of Circenis, OpenPGP and IBM teach a system including the elements of claim 1 wherein the system further includes a mechanism for receiving an input from an end-user that initiates logging of log entries into the log file each time logging is desired by the user. *(Paragraph [0019] teaches a mechanism for receiving an input from an end-user that initiates logging of log entries, as shown by the "pay per use metering application that collects metrics data associated with computer system)* The user can use the system as many times as desired. The logger logs each use of the system. Therefore logging is initiated as desired by the end user.

Regarding Claim 8,

The combined references of Circenis, OpenPGP and IBM a system including the elements of claim 1 wherein the system further includes an initializing mechanism for determining each instance logging is to begin and initiating logging of log entries only in response to that initializing mechanism. *("The iCOD computer could save usage data to a log file or a central metering device" Paragraph [0024] Circenis) ("an iCOD computer residing on an isolated site should be designed to discourage any reverse engineering or other tampering and to make such tampering evident to the iCOD computer vendor" Paragraph*

[0023] Circenis) The Examiner interprets the iCOD inherently having an initializing mechanism. The Examiner interprets the design to discourage tampering as so that only logging entries are only initiated in response to the initializing mechanism.

Regarding Claim 9,

The combined references of Circenis OpenPGP, and IBM teach a system including the elements of claim 1 wherein the system uses a public key to encrypt the log entry which has been created and a private key corresponding to the public key is used to decrypt the log which has been created at a secure location. *("Public and private encryption/decryption key pairs where data encrypted by a public key can only be decrypted with a corresponding private key, and visa versa, provide data confidentiality" Paragraph [0025], Figure 4 of Circenis shows encryption and Figure 5 shows decryption)*

Regarding Claim 10

Circenis teaches a method for diagnosing software in a tamper-resistant environment comprising the steps of:

monitoring at least one software operation activity within the tamper-resistant environment and generating messages in response to at least one instance of software execution within the tamper-resistant environment, wherein the software operation activity is identified and selected by an end-user to be monitored, wherein the end-user is a user that initiates execution of the software at a system associated with the end user; *("Using the tamper-evident system 200 of Fig. 3, a sender is able to monitor and control application utilization by collecting data associated with the application, creating tamper-evident data records, and providing the tamper-evident data records" Paragraph [0037])*

logging at least one software operation activity relating to a generated message by replacing a random data with an encrypted record of the software operation activity;

*(Figure 4 teaches encrypting the log entry for at least one set of data, particularly **step 320** "Sign data entry with application private key", **step 325** "Encrypt with vendor public key" and **step 330** "Store in data log")*

and sending the log file to a secure location where it the log file can be decrypted and analyzed; *("The data log may also be transmitted to a remote system (comprising, for example, the validation computer 150) over a network connection" Paragraph [0043] of Circenis, Figure 3 shows the transmission of the log file 115 to the secure processing location away from the system 150)*

and analyzing the decrypted log file data and providing information on the operation of the software in the tamper-resistant environment. *("The use of the vendor public and private keys ensures that only the vendor can decrypt the data logon the computer system…to preserve the confidentiality of the data log" Paragraph [0043]) It is inherent that the data log will provide information on the operation of the software in the tamper-resistant environment.*

The Examiner interprets the processor that monitors software execution, wherein the one instance is identified by an end-user to be monitored by the processor as the user executing software on a device wherein each instance of execution is to be monitored by the processor, and in response to the one instance being executed a log entry is created (See Paragraph [0019] where Circenis teaches the pay per use application collecting metrics data to a data log).

Circenis does not explicitly teach an encryption system which generates at least one symmetric key and encrypts the log entry for the at least one set of data using the symmetric key, wherein the encryption system encrypts the symmetric key using a public key associated with the encryption system, wherein the log file includes the symmetric key which has been encrypted with the public key.

PGP ("Pretty Good Privacy") is a program that provides cryptographic privacy and authentication, and was created by Phillip Zimmermann in 1991. The OpenPGP standard (1998) is cited, but any PGP product teaches the generic method of:

1. Creating a message
2. Generating a symmetric key to be used as a session key for the message
3. Encrypting the session key using each recipient's public key. These "encrypted session keys" start the message.
4. The sending PGP encrypts the message using the session key, which forms the remainder of the message.
5. The receiving PGP device decrypts the session key using the recipient's private key
6. The receiving PGP decrypts the message using the session key.

Because Circenis already teaches one method of encrypting the data log, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the public-private key encryption of Circenis with the well known method of PGP, where the symmetric key is generated, the log entry is encrypted using the symmetric key, a public key encrypts the symmetric key, and the log file includes the symmetric key which has been encrypted with the public key.

The motivation is that PGP provides a more secure way of encrypting the log entries.

Circenis and OpenPGP do not explicitly teach

turning on logging and establishing a pointer for a location of a next logged software operation activity;

moving the pointer when a log entry has been made to a next available log position;

wrapping the pointer to a beginning of the log file when the log file is full of log entries;

IBM teaches turning on logging and establishing a pointer for a location of a next logged software operation activity; moving the pointer when a log entry has been made to a next available log position; *("The alog command can maintain and manage logs. It reads standard input, writes to standard output, and copies the output into a fixed-size file. This file is treated as a circular log" Section 2.4.1) It is inherent that a circular log has a pointer that moves to the next logged software operation activity.*

wrapping the pointer to a beginning of the log file when the log file is full of log entries; *("If the file is full, new entries are written over the oldest existing entries" Section 2.4.1). It is inherent that a circular log will wrap around and fill the log file from a beginning when the log file has been filled.*

The combined references of Circenis and IBM do not further teach generating a log file full of random data;

It would have been obvious to one of ordinary skill in the art at the time of the invention to insert random data into the log file when it is initially created.

The motivation is that it is inherent that the circular log is of a fixed size so it must be initialized with some values. One of ordinary skill in the art would know to initialize the circular log with random values.

Regarding Claim 11,

Circenis, OpenPGP and IBM teach a method including the steps of claim 10 wherein the step of turning on logging includes the steps of receiving an user input that logging is desired and initiating the logging in response thereto. *("The iCOD computer could save usage data to a log file or a central metering device that a vendor employee could check periodically by visiting the site." Paragraph [0024] Circenis) The Examiner interprets the vendor employee as the user the indicates logging is desired)*

Regarding Claim 12,

Circenis, OpenPGP and IBM teach a method including the steps of claim 10 wherein the step of at least one software operation activity further includes the steps of determining whether the software operation activity is to be logged, *The Examiner interprets that before the data is logged, inherently, there must be a step of determining whether the activity is to be logged.*

and if so, determining when to encrypt the software operation activity to obscure what is being logged. *("Encryption may be added to keep the customer's data log confidential" Paragraph [0039] Circenis) The Examiner interprets that before the data log is encrypted there must inherently be a determining step of when to encrypt the software activity.*

Regarding Claim 13,

Circenis, OpenPGP and IBM teach a method including the steps of claim 10 wherein the step of logging the software operation activity further includes the steps of determining a next available log position, *It is inherent that a circular log requires determing a next available log position.*

replacing existing data in the location with the data from the software operation activity, *("If the file is full, new entries are written over the oldest existing entries" Section 2.4.1, IBM).*

and updating the pointer to provide a location of the next logged software operation activity. *It is inherent that a circular log updates the pointer to provide a location of the next activity.*

Regarding Claim 14,

Circenis, OpenPGP and IBM teach a method including the steps of claim 10 and further including the step of receiving a command from a user that indicates that sending the log file to a remote location is desired and transmitting the log file in response thereto. *("Fig. 5 is a flowchart illustrating steps in validating the data. The program starts (step 355) and the data log is copied to the validation computer through an intermediary device or medium (step 360)" Circenis) Before the data can be validated there must inherently be some indication for the log file to be transmitted.*

Regarding Claim 15,

Circenis teaches a method of analyzing the operation of software in a remote protected processing environment, the method including:

receiving from the remote protected processing environment an encrypted log file comprising at least one log entry with at least one set of data derived from at least one instance of software execution monitored in response to a user identifying and selecting the one instance of software execution, wherein the end-user is a user that initates execution of the software at a system associated with the end-user, whereby the set of data is used to diagnose the software execution; *("The data log also may be transmitted to a remote system (comprising, for example, the validation computer) over a network connection" Paragraph [0043])* determining a decrypting key for the encrypted log file and decrypting the encrypted log file; *("The software on the validation computer may then decrypt each of the data log entries in the data log using the vendor private key" Paragraph [0043])*

analyzing the log entry at the remote protected processing environment and to determine whether an operation of the remote protected processing environment corresponding to the at least one set of data derived from at least one instance of software execution is appropriate; *("The data log is then further inspected by the vendor for evidence of customer tampering." Paragraph [0044])*

and reporting the results of the analyzing step. *(The Examiner interprets the vendor inspecting the data logs as reporting the results of the analyzing step)*

The Examiner interprets the processor that monitors software execution, wherein the one instance is identified by an end-user to be monitored by the processor as the user executing software on a device wherein each instance of execution is to be monitored by the processor, and in response to the one instance being executed a log entry is created (See Paragraph [0019] where Circenis teaches the pay per use application collecting metrics data to a data log).

Circenis does not explicitly teach an encryption system which generates at least one symmetric key and encrypts the log entry for the at least one set of data using the symmetric key, wherein the encryption system encrypts the symmetric key using a public key associated with the encryption system, wherein the log file includes the symmetric key which has been encrypted with the public key, determining a private decrypting key corresponding to the public key associated with the system, and using the decrypting key and the private decrypting key.

PGP ("Pretty Good Privacy") is a program that provides cryptographic privacy and authentication, and was created by Phillip Zimmermann in 1991. The OpenPGP standard (1998) is cited, but any PGP product teaches the generic method of:

1. Creating a message
2. Generating a symmetric key to be used as a session key for the message
3. Encrypting the session key using each recipient's public key. These "encrypted session keys" start the message.
4. The sending PGP encrypts the message using the session key, which forms the remainder of the message.
5. The receiving PGP device decrypts the session key using the recipient's private key
6. The receiving PGP decrypts the message using the session key.

Because Circenis already teaches one method of encrypting the data log, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the public-private key encryption of Circenis with the well known method of PGP, where the symmetric key is generated, the log entry is encrypted using the symmetric key, a public key encrypts the symmetric key, and the log file includes the symmetric key which has been encrypted with the public key.

The motivation is that PGP provides a more secure way of encrypting the log entries.

Circenis and OpenPGP does not explicitly teach that the data log is of substantially-constant size

IBM teaches that the data log is of substantially-constant size.
("The alog command can maintain and manage logs. It reads standard input , writes to standard output, and copies the output into a fixed-size file. This file is treated as a circular log" Section 2.4.1)

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the data log monitoring system of Circenis with the fixed-sized log (circular log) of IBM.

The motivation is that the circular log is well known in the art and without much modification the circular log can be used in the system of Circenis with no difference in result.

Regarding Claim 16,

Circenis, OpenPGP and IBM teach a method providing the steps of claim 15. It is inherent that before "the data log…may be transmitted to a remote system" (Paragraph

[0043] *Circenis*) that an instruction to send the encrypted log file to the remote location is needed.

Circenis teaches including providing an instruction to initiate a logging of messages each time logging is desired by the user *("The iCOD computer could save usage data to a log file or a central metering device that a vendor employee could check periodically by visiting the site." Paragraph [0024]) The Examiner interprets the vendor employee as the user the indicates logging is desired)*

Regarding Claim 17,

Circenis, OpenPGP and IBM teach a method providing the steps of claim 16.

Circenis, OpenPGP and IBM do not explicitly teach wherein the instruction to initiate logging of messages includes the step of initiating programming within the remote protected processing environment to replace information in the encrypted log file with encrypted information relating to the operation of the remote protected processing environment.

It would have been obvious to one of ordinary skill in the art at the time of the invention to include programming within the remote system to replace information in the encrypted file log with encrypted information relating to the operation of the remote protected system.

The motivation is that in the system of Circenis, once the data log is passed to the remote system, it is in the hands of the vendor or system administrator. Because tampering is no longer an issue the vendor can adjust the data log to include whatever instruction is deemed necessary. One of ordinary skill in the art would be able to replace encrypted data log information with encrypted information relating to the operation of the remote protected system.

Regarding Claim 18,

Circenis, OpenPGP and IBM teach a method providing the steps of claim 17.

Circenins and IBM do not explicitly teach wherein the step of replacing information in the encrypted log file includes the step of replacing random data which was placed in the encrypted log file when it was created.

It would have been obvious to one of ordinary skill in the art at the time of the invention to insert random data into the log file when it is initially created.

The motivation is that it is inherent that the circular log is of a fixed size so it must be initialized with some values. One of ordinary skill in the art would know to initialize the circular log with random values.

Regarding Claim 19,

Circenis, OpenPGP and IBM teach a method providing the steps of claim 17. IBM teaches a circular log wherein the step of replacing information in the log file inherently includes the step of using a pointer to a next location in the log file and the pointer wraps to a beginning the log file after the encrypted log file has been filled.

Regarding Claim 20,

Circenis teaches a computer program product for analyzing software running in a tamper-resistant environment, the computer program product comprising instructions for:

at least one set of data serviced from at least one instance of software execution identified and selected by an end-user to be monitored whereby the set of data is used to diagnose the software execution wherein the end-user is a user that initates execution of the software at a system associated with the end-user; *("Using the tamper-evident system 200 of Fig. 3, a sender is able to monitor and control application utilization by collecting data associated with the application, creating tamper-evident data records, and providing the tamper-evident data records" Paragraph [0037])*

encrypting the recording of the at least one set of data using a key; *(Figure 4 teaches encrypting the log entry for at least one set of data, particularly **step 320** "Sign data entry with application private key", **step 325** "Encrypt with vendor public key" and **step 330** "Store in data log")*

responding to a command and sending the encrypted log file comprising the at least one set of data which has been encrypted and sequentially recoded in the storage block to a remote location for decryption and analysis. *("The data log may also be* *transmitted to a remote system (comprising, for example, the validation computer 150) over a network connection" Paragraph [0043] of Circenis, Figure 3 shows the transmission of the log file 115 to the secure processing location away from the system 150). The Examiner interprets the data in the log of Circenis as being sequentially recoded. ("The sequence numbers of the data log entries are also checked for gaps or data log entries that are out of sequence" Paragraph [0044])*

The Examiner interprets the processor that monitors software execution, wherein the one instance is identified by an end-user to be monitored by the processor as the user executing software on a device wherein each instance of execution is to be monitored by the processor, and in response to the one instance being executed a log entry is created (See Paragraph [0019] where Circenis teaches the pay per use application collecting metrics data to a data log).

Circenis does not explicitly teach an encryption system which generates at least one symmetric key and encrypts the log entry for the at least one set of data using the symmetric key, wherein the encryption system encrypts the symmetric key using a public key associated with the encryption system, wherein the log file includes the symmetric key which has been encrypted with the public key.

PGP ("Pretty Good Privacy") is a program that provides cryptographic privacy and authentication, and was created by Phillip Zimmermann in 1991. The OpenPGP standard (1998) is cited, but any PGP product teaches the generic method of:

1. Creating a message

2. Generating a symmetric key to be used as a session key for the message

3. Encrypting the session key using each recipient's public key. These "encrypted session keys" start the message.

4. The sending PGP encrypts the message using the session key, which forms the remainder of the message.

5. The receiving PGP device decrypts the session key using the recipient's private key

6. The receiving PGP decrypts the message using the session key.

Because Circenis already teaches one method of encrypting the data log, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the public-private key encryption of Circenis with the well known method of PGP, where the symmetric key is generated, the log entry is encrypted using the symmetric key, a public key encrypts the symmetric key, and the log file includes the symmetric key which has been encrypted with the public key.

The motivation is that PGP provides a more secure way of encrypting the log entries.

Circenis and OpenPGP do not explicitly teach

recording at least one set of data, which has been encrypted sequentially in a storage block of a substantially fixed size;

maintaining a pointer to a next available location for recording the at least one set of data sequentially in the storage block;

IBM teaches recording at least one set of data, which has been encrypted in a storage block of a substantially fixed size; *("The alog command can maintain and manage logs. It reads standard inpu , writes to standard output, and copies the output into a fixed-size file. This file is treated as a circular log" Section 2.4.1)*

It is inherent that a circular log maintains a pointer to a next available location for recording the at least one set of data sequentially in the storage block;

It would have been obvious to one of ordinary skill in the art at the time of the invention to modify the monitoring system of Circenis to store the encrypted log entries in a circular log as described by IBM.

The motivation is that a circular log is a well known way to store a log file, where the circular log is inherently of a fixed size. It is inherent that a circular log will contain at least a pointer which identifies the next storage location for a next log entry.

Regarding Claim 21,

Circenis, OpenPGP and IBM teach the computer program product of claim 20. Circenis and IBM do not further teach instructions for:

Initializing the storage block of a substantially fixed size with random information which has been encrypted to provide a block of apparent data.

It would have been obvious to one of ordinary skill in the art at the time of the invention to insert random data into the log file when it is initially created.

The motivation is that it is inherent that the circular log is of a fixed size so it must be initialized with some values. One of ordinary skill in the art would know to initialize the circular log with random values.

Regarding Claim 22,

Circenis, OpenPGP and IBM he computer program product of claim 20, further comprising instructions for:

writing the at least one set of data which has been encrypted and recorded events in a sequential order in the storage block *("The sequence numbers of the data log entries are also checked for gaps or data log entries that are out of sequence…Inconsistencies in…the sequence numbers would provide evidence of tampering with the data log" Paragraph [0044] of Circenis). Because the data log is supposed to be sequential, the Examiner interprets that the data is written in a sequential order.*

In a circular log it is inherent for wrapping around when an end of the storage block of the substantially fixed-size memory is reached.

Conclusion

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to HARRIS C. WANG whose telephone number is (571)270-1462. The examiner can normally be reached on M-F 9-5:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, KRISTINE KINCAID can be reached on (571) 272-4063. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

HCW
/Kristine Kincaid/
Supervisory Patent Examiner, Art Unit 2139